Stable Torque Optimization for Redundant Robots using a Short Preview

Khaled Al Khudir Gaute Halvorsen Leonardo Lanari Alessandro De Luca

***Abstract*—We consider the known phenomenon of torque oscillations and motion instabilities that occur in redundant robots during the execution of sufficiently long Cartesian trajectories when the joint torque is instantaneously minimized. In the framework of on-line local redundancy resolution methods, we propose basic variations of the minimum torque scheme to address this issue. Either the joint torque norm is minimized over two successive discrete-time samples using a short preview window, or we minimize the norm of the difference with respect to a desired momentum-damping joint torque, or the two schemes are combined together. The resulting local control methods are all formulated as well-posed linear-quadratic problems, and their closed-form solutions generate also low joint velocities while addressing the primary torque optimization objectives. Stable and consistent behaviors are obtained along short or long Cartesian position trajectories, as illustrated with simulations on a 3R planar arm and with experiments on a 7R KUKA LWR robot.**

***Index Terms*—Optimization and Optimal Control, Redundant Robots, Motion Control, Dynamics.**

I. Introduction

REDUCING the torque needed to execute a given robot task is a common control objective where dynamics plays a major role. When the desired Cartesian task is constraining the robot end-effector to a m-dimensional trajectory and the number of its actuated degrees of freedom is $n > m$, the robot will be redundant with respect to the task. In this case, the authors in [1] presented the first method for torque optimization, obtaining the solution in closed form that instantaneously minimizes the norm of the joint torque, while executing a desired Cartesian trajectory. However, for longer movements, this local optimization method often exhibits an unstable behavior in the form of motion oscillations, growing joint velocities, and sudden whipping torque effects with loss of control in practice. The obvious countermeasure would be to pursue a global torque optimization over the whole motion trajectory [2], [3], but this implies the off-line numerical solution of a two-point boundary value problem and the availabillity of the entire desired task in advance.

Wishing instead to keep an on-line redundancy resolution method, which is also suited for conversion into a sensor-based feedback control scheme, local optimization alternatives have been further explored. The minimization of the infinity-norm of the joint torque was proposed in [4], but the torque explosion problem would still appear and extra inequality constraints had to be added to the algorithm. A joint decomposition formulation was presented in [5] for torque optimization, with the aim of forcing a return to zero of the joint velocities at the end of the motion task. The relevance of keeping the joint velocity under control in order to address the instability problem was pointed out also in [6]. Special attention was given to the robot self-motion dynamics, providing also a first explanation for the good torque performance experienced in simulations when using a purely kinematic joint velocity minimization scheme [7]. Based on this, a balancing between the minimum velocity and the minimum torque solutions was proposed in [8]. The instabilities that can occur when redundancy is solved at the second-order level (in terms of joint accelerations or torques) were analyzed in [9], relating them to the smallest singular value of the task Jacobian and to particular null-space accelerations that cause a quadratic growth of joint velocities. In [10], a complete characterization of the instability phenomenon was given, studying the self-motion manifold in the case of robots with one degree of redundancy. Interestingly, it was shown that simple damping of joint velocities in the null space of the task may not prevent large motion oscillations in the long run.

Unfortunately, the above mentioned local methods do not provide conclusive answers: instability problems are still encountered for longer movements, or they would fail for certain tasks and initial conditions, or countermeasures were proven to work only in relatively simple cases (i.e., when $n - m = 1$). Last but not least, most papers on model-based optimization of joint torques in redundant robots have presented only simulation results on simple manipulators. Among the few exceptions, there are [11] and lately [12], [13], where, however, the issue of instability in local torque minimization on long trajectories is not explicitly addressed.

In this paper, we introduce two basic variations to the minimum torque norm (*MTN*) scheme of [1] that address the issue of unstable joint motion in redundant robots. In the first scheme, we propose the minimization of the joint torque norm over two successive discrete-time samples using a short preview window (possibly, in the next sampling instant). Suitable dynamic approximations are introduced in the estimation of the future (proximal) robot state, so as to keep a linear-quadratic (LQ) formulation for the problem, in a way similar to [14]. Such a dynamic optimization scheme, that we denote model-based preview (*MBP*), can be seen as a compromise between local and global redundancy resolution methods, trying to inherit the best of both worlds —real-time

Manuscript received: September 10, 2018; revised December 26, 2018; accepted January 21, 2019. This paper was recommended for publication by Editor P. Rocco upon evaluation of the Associate Editor and Reviewers' comments.

The authors are with the Dipartimento di Ingegneria Informatica, Automatica e Gestionale, Sapienza Università di Roma, Via Ariosto 25, 00185 Roma, Italy (e-mail: alkhudir@diag.uniroma1.it; gaudaost@gmail.com; lanari@diag.uniroma1.it; deluca@diag.uniroma1.it).

Digital Object Identifier (DOI): see top of this page.

simplicity and stable behavior.

Because of the presence of a predictive window in the future, *MBP* is similar to a model predictive control (*MPC*) approach. However, most linear [15] and nonlinear [16] *MPC* methods in robotics have not considered explicitly robot redundancy or on-line task constraints. When real-time execution is a must, different local approximations are being performed to reduce the computational effort (e.g., using a linear inverted pendulum predictive model in humanoid gait control [17]). A main difference with respect to *MPC* is that *MBP* does not need to compute/predict multiple system samples in a future window. Rather, in its basic version our method uses just a single preview state, which may be placed close in time or further away from the current one, where the task-based equality constraint is also imposed. Computational efficiency is achieved even for a large number of degrees of freedom, thanks to the closed-form solution of a linear-quadratic (*LQ*) problem that is always guaranteed to be positive definite. On the other hand, contrary to the common case in *MPC*, bounds on the future state are not considered. While feasible in principle, such extension is in fact not strictly needed because of the nature of the problem addressed, i.e., stable optimization of the motion torques.

In the second proposed control scheme, we choose the command torque that realizes the Cartesian task and is closest in norm to a desired torque, which is proportional to the current generalized momentum of the robot. This induces a natural dynamic damping of the joint velocities, preventing their oscillations or growth, and we label the solution as minimum torque norm with damping (*MTND*). The two proposed modifications can also be combined, leading to a model-based preview scheme with damping (*MBPD*). Again, these schemes are formulated as well-posed *LQ* problems, providing efficiently the solution in closed form. The performance of the new controllers was tested in a large number of short and long motions, always yielding a stable behavior under torque optimization.

The rest of the paper is organized as follows. The general formulation of the instantaneous torque optimization as a *LQ* problem is reviewed is Sec. II. Section III presents the derivation of the proposed optimization method with model-based preview, as well as the damping extensions (Sec. III-A). Sec. IV-A reports comparative numerical results for a 3R planar arm performing two-dimensional trajectories of different length. Experimental results are reported in Sec. V for a 7R KUKA LWR4 robot executing two exemplary positional tasks (also shown in the accompanying video). Conclusions and future work are summarized in Sec. VI.

II. Instantaneous Minimum Torque Solution

Consider a robot manipulator with (generalized) joint coordinates $\boldsymbol{q} \in \mathbb{R}^n$, described by the dynamic model,

$$\boldsymbol{M}(\boldsymbol{q})\ddot{\boldsymbol{q}} + \boldsymbol{c}(\boldsymbol{q},\dot{\boldsymbol{q}}) + \boldsymbol{g}(\boldsymbol{q}) = \boldsymbol{\tau}, \tag{1}$$

with symmetric, positive definite inertia matrix $\boldsymbol{M}$, Coriolis and centrifugal terms $\boldsymbol{c}$ (with quadratic dependence on $\dot{\boldsymbol{q}}$), gravity term $\boldsymbol{g}$, and commanded joint torque $\boldsymbol{\tau}$. The components $c_i(\boldsymbol{q},\dot{\boldsymbol{q}})$, for $i = 1,\dots,n$, of vector $\boldsymbol{c}$ are given by

$$c_i = \dot{\boldsymbol{q}}^T \boldsymbol{C}_i(\boldsymbol{q})\dot{\boldsymbol{q}}, \quad \boldsymbol{C}_i = \frac{1}{2}\left(\left(\frac{\partial \boldsymbol{m}_i}{\partial \boldsymbol{q}}\right) + \left(\frac{\partial \boldsymbol{m}_i}{\partial \boldsymbol{q}}\right)^T - \frac{\partial \boldsymbol{M}}{\partial q_i}\right), \tag{2}$$

where $\boldsymbol{m}_i(\boldsymbol{q})$ is the ith column of $\boldsymbol{M}$ and the elements C_{ijk} of matrix $\boldsymbol{C}_i$ are the so-called Christoffel symbols. A convenient factorization of $\boldsymbol{c}$ is given by

$$\boldsymbol{c}(\boldsymbol{q},\dot{\boldsymbol{q}}) = \boldsymbol{S}(\boldsymbol{q},\dot{\boldsymbol{q}})\,\dot{\boldsymbol{q}} = \mathrm{col}\left\{\dot{\boldsymbol{q}}^T \boldsymbol{C}_i(\boldsymbol{q})\right\}\dot{\boldsymbol{q}}, \tag{3}$$

in which matrix $\boldsymbol{S}$ yields skew-symmetry for $\dot{\boldsymbol{M}} - 2\boldsymbol{S}$. In the following, we will use also the compact notation

$$\boldsymbol{n}(\boldsymbol{q},\dot{\boldsymbol{q}}) = \boldsymbol{c}(\boldsymbol{q},\dot{\boldsymbol{q}}) + \boldsymbol{g}(\boldsymbol{q}) \tag{4}$$

to denote all terms in (1) not depending on joint acceleration.

The robot should perform a task described by the variables $\boldsymbol{x} \in \mathbb{R}^m$, with $m < n$, related to $\boldsymbol{q}$ by the task kinematics $\boldsymbol{x} = \boldsymbol{f}(\boldsymbol{q})$. The differential relations at the first- and second-order level are given by

$$\dot{\boldsymbol{x}} = \frac{\partial \boldsymbol{f}(\boldsymbol{q})}{\partial \boldsymbol{q}}\,\dot{\boldsymbol{q}} = \boldsymbol{J}(\boldsymbol{q})\dot{\boldsymbol{q}}, \tag{5}$$

where $\boldsymbol{J}$ the $m \times n$ task Jacobian, and respectively by

$$\ddot{\boldsymbol{x}} = \boldsymbol{J}(\boldsymbol{q})\ddot{\boldsymbol{q}} + \dot{\boldsymbol{J}}(\boldsymbol{q})\dot{\boldsymbol{q}} = \boldsymbol{J}(\boldsymbol{q})\ddot{\boldsymbol{q}} + \boldsymbol{h}(\boldsymbol{q},\dot{\boldsymbol{q}}). \tag{6}$$

Note that vector $\boldsymbol{h}$ has a quadratic dependence on $\dot{\boldsymbol{q}}$. The redundancy of the robot with respect to the task can be resolved at the acceleration level by minimizing some (possibly, weighted) norm of the joint torques. For this, a standard linear-quadratic optimization problem can be defined as

$$\min_{\ddot{\boldsymbol{q}}^*} H = \frac{1}{2}\left\|\boldsymbol{\tau}^*\right\|^2 = \frac{1}{2}\ddot{\boldsymbol{q}}^{*T}\boldsymbol{Q}\ddot{\boldsymbol{q}}^* + \boldsymbol{r}^T\ddot{\boldsymbol{q}}^* + s \tag{7a}$$

$$\text{s.t. } \boldsymbol{\tau}^* = \boldsymbol{Q}^{\frac{1}{2}}\ddot{\boldsymbol{q}}^* + \boldsymbol{Q}^{-\frac{1}{2}}\boldsymbol{r} \tag{7b}$$

$$\boldsymbol{b} = \boldsymbol{A}\ddot{\boldsymbol{q}}^*, \tag{7c}$$

assuming that $\boldsymbol{Q}$ is a positive definite (weighting) matrix, matrix $\boldsymbol{A}$ has full (row) rank, $\boldsymbol{b} \in \mathbb{R}^n$, $\boldsymbol{r} \in \mathbb{R}^n$, and s is a scalar. All torque optimization problems in the paper can be formulated as (7) for specific choices of $\boldsymbol{Q}$, $\boldsymbol{A}$, $\boldsymbol{b}$ and $\boldsymbol{r}$ (while s is irrelevant). Equation (7c) represents the task (kinematic) constraint, while (7b) simply provides the command torque $\boldsymbol{\tau}^*$ from the dynamic model equation (1) when the joint acceleration solution $\ddot{\boldsymbol{q}}^*$ is being used.

The unique solution to (7) is obtained with the method of Lagrange multipliers [18] as

$$\ddot{\boldsymbol{q}}^* = \boldsymbol{A}_{\boldsymbol{Q}}^{\#}\left(\boldsymbol{b} + \boldsymbol{A}\boldsymbol{Q}^{-1}\boldsymbol{r}\right) - \boldsymbol{Q}^{-1}\boldsymbol{r}, \tag{8}$$

with the weighted pseudoinverse

$$\boldsymbol{A}_{\boldsymbol{Q}}^{\#} = \boldsymbol{Q}^{-1}\boldsymbol{A}^T\left(\boldsymbol{A}\boldsymbol{Q}^{-1}\boldsymbol{A}^T\right)^{-1}.$$

Note that the solution (8) can be written also as

$$\ddot{\boldsymbol{q}}^* = \boldsymbol{A}_{\boldsymbol{Q}}^{\#}\,\boldsymbol{b} - \left(\boldsymbol{I} - \boldsymbol{A}_{\boldsymbol{Q}}^{\#}\,\boldsymbol{A}\right)\boldsymbol{Q}^{-1}\boldsymbol{r} \tag{9}$$

to emphasize the presence of a control action in the null-space of the task matrix $\boldsymbol{A}$, represented by the term $\boldsymbol{Q}^{-1}\boldsymbol{r}$.

Let a sequence of control instants $t_k = kT_s$, $k = 0, 1, \ldots$, be given for a sampling time $T_s > 0$. Denote by $\boldsymbol{q}_k = \boldsymbol{q}(t_k)$ and $\dot{\boldsymbol{q}}_k = \dot{\boldsymbol{q}}(t_k)$ the position and velocity (i.e., the current state) of the robot at time $t = t_k$. We can then select the acceleration $\ddot{\boldsymbol{q}}_k$ at t_k (equivalently, the torque $\boldsymbol{\tau}_k$) by solving the problem

$$\begin{aligned} \min_{\ddot{\boldsymbol{q}}_k} H_1 \;&= \tfrac{1}{2}\,\|\boldsymbol{\tau}_k\|^2 \\ \text{s.t.}\quad \boldsymbol{\tau}_k &= \boldsymbol{M}_k\ddot{\boldsymbol{q}}_k + \boldsymbol{n}_k \\ \ddot{\boldsymbol{x}}_k &= \boldsymbol{J}_k\ddot{\boldsymbol{q}}_k + \boldsymbol{h}_k, \end{aligned} \tag{10}$$

where we have used the shorthand notations

$$\begin{aligned} \boldsymbol{M}_k &= \boldsymbol{M}(\boldsymbol{q}_k), & \boldsymbol{n}_k &= \boldsymbol{n}(\boldsymbol{q}_k, \dot{\boldsymbol{q}}_k), \\ \boldsymbol{J}_k &= \boldsymbol{J}(\boldsymbol{q}_k), & \boldsymbol{h}_k &= \boldsymbol{h}(\boldsymbol{q}_k, \dot{\boldsymbol{q}}_k). \end{aligned}$$

Problem (10) is in the form (7) with the substitutions

$$\begin{aligned} \boldsymbol{Q} = \boldsymbol{M}_k^2, \quad \boldsymbol{r} = \boldsymbol{M}_k\boldsymbol{n}_k, \quad s = \tfrac{1}{2}\boldsymbol{n}_k^T\boldsymbol{n}_k, \\ \boldsymbol{A} = \boldsymbol{J}_k, \qquad \boldsymbol{b} = \ddot{\boldsymbol{x}}_k - \boldsymbol{h}_k. \end{aligned} \tag{11}$$

Thus, the optimal acceleration $\ddot{\boldsymbol{q}}^* = \ddot{\boldsymbol{q}}_k$ is given by (8) and the associated optimal torque $\boldsymbol{\tau}^* = \boldsymbol{\tau}_k$ is then obtained from (7b). Using (9), we see that the null-space action that minimizes the joint torque norm is given by $\boldsymbol{M}_k^{-1}\boldsymbol{n}_k$. This is indeed the solution found in [1] and denoted here as *MTN*.

For a given Cartesian task $\boldsymbol{x} = \boldsymbol{x}_d(t)$, the direct use of $\ddot{\boldsymbol{x}}_k = \ddot{\boldsymbol{x}}_{d,k} = \ddot{\boldsymbol{x}}_d(t_k)$ in the expression of $\boldsymbol{b}$ in (11) provides an open-loop solution. In order to reduce the Cartesian tracking error during motion, a stabilizing PD feedback control term is inserted at the task level in $\boldsymbol{b}$, obtaining

$$\boldsymbol{b} = \ddot{\boldsymbol{x}}_{d,k} + \boldsymbol{K}_D(\dot{\boldsymbol{x}}_{d,k} - \dot{\boldsymbol{x}}_k) + \boldsymbol{K}_P(\boldsymbol{x}_{d,k} - \boldsymbol{x}_k) - \boldsymbol{h}_k \tag{12}$$

with $m \times m$ (diagonal) gain matrices $\boldsymbol{K}_D > 0$ and $\boldsymbol{K}_P > 0$.

III. Model-based preview of evolution

To prevent a long term unstable behavior of the solution that instantaneously minimizes torque we will include an estimate of the future robot state in the optimization process. At time $t = t_k$, consider a preview window $T = pT_s$, for some integer $p \geq 1$. At the sampling instant $t_{k+p} = t_k + T$, we shall associate a suitable approximation of the robot state. In the following, we shall consider that $T = T_s$ (or $p = 1$), but the same formulas will be used also as an approximated model for a larger $T > 0$. Suppose that a constant acceleration $\ddot{\boldsymbol{q}}_k$ is applied during a preview window $T = T_s$ in the time interval $[t_k, t_{k+1})$. The following discrete-time evolution holds then exactly for the robot state

$$\begin{aligned} \dot{\boldsymbol{q}}_{k+1} &= \dot{\boldsymbol{q}}(t_{k+1}) = \dot{\boldsymbol{q}}_k + \ddot{\boldsymbol{q}}_k\,T, \\ \boldsymbol{q}_{k+1} &= \boldsymbol{q}(t_{k+1}) = \boldsymbol{q}_k + \dot{\boldsymbol{q}}_k T + \tfrac{1}{2}\,\ddot{\boldsymbol{q}}_k\,T^2. \end{aligned} \tag{13}$$

We can indeed associate from (1) a unique $\boldsymbol{\tau}_k$ to any $\ddot{\boldsymbol{q}}_k$.

Similarly to (10), which is in fact the no preview solution for $T = 0$, we formulate the following optimization problem

$$\begin{aligned} \min_{\ddot{\boldsymbol{q}}_k, \ddot{\boldsymbol{q}}_{k+1}} H_2 \;&= \tfrac{1}{2}\left(\omega_k\|\boldsymbol{\tau}_k\|^2 + \omega_{k+1}\|\boldsymbol{\tau}_{k+1}\|^2\right) \\ \text{s.t.}\quad \boldsymbol{\tau}_k &= \boldsymbol{M}_k\ddot{\boldsymbol{q}}_k + \boldsymbol{n}_k \\ \ddot{\boldsymbol{x}}_k &= \boldsymbol{J}_k\ddot{\boldsymbol{q}}_k + \boldsymbol{h}_k \\ \boldsymbol{\tau}_{k+1} &= \boldsymbol{M}_{k+1}\ddot{\boldsymbol{q}}_{k+1} + \boldsymbol{n}_{k+1} \\ \ddot{\boldsymbol{x}}_{k+1} &= \boldsymbol{J}_{k+1}\ddot{\boldsymbol{q}}_{k+1} + \boldsymbol{h}_{k+1}, \end{aligned} \tag{14}$$

where we have used the notations

$$\begin{aligned} \boldsymbol{M}_{k+1} &= \boldsymbol{M}(\boldsymbol{q}_{k+1}), & \boldsymbol{n}_{k+1} &= \boldsymbol{n}(\boldsymbol{q}_{k+1}, \dot{\boldsymbol{q}}_{k+1}), \\ \boldsymbol{J}_{k+1} &= \boldsymbol{J}(\boldsymbol{q}_{k+1}), & \boldsymbol{h}_{k+1} &= \boldsymbol{h}(\boldsymbol{q}_{k+1}, \dot{\boldsymbol{q}}_{k+1}). \end{aligned}$$

For generality, we introduced in (14) also the two constants $\omega_k \geq 0$ and $\omega_{k+1} \geq 0$ (with $\omega_k^2 + \omega_{k+1}^2 \neq 0$) that relatively weigh the torque norms at the current and preview instants.

When using (13) within the nonlinear dynamic terms in (14), the formulation loses the original structure of a *LQ* problem in the unknown joint accelerations. The constraints in (14) will still be linear in the unknown $\ddot{\boldsymbol{q}}_{k+1}$, but become nonlinear in the unknown $\ddot{\boldsymbol{q}}_k$. Moreover, the objective function will no longer be quadratic in the $\ddot{\boldsymbol{q}}_k$. This makes problem (14) impossible to solve in a closed form, as opposed to (10). On the other hand, if one 'freezes' the dependencies of $\boldsymbol{M}_{k+1}$, $\boldsymbol{J}_{k+1}$, $\boldsymbol{n}_{k+1}$, and $\boldsymbol{h}_{k+1}$ to their value at time $t = t_k$, or even removes the dependence of these terms from $\ddot{\boldsymbol{q}}_k$, the formulation (14) would become *separable* in two independent sub-problems, one depending only on $\ddot{\boldsymbol{q}}_k$, the other only on $\ddot{\boldsymbol{q}}_{k+1}$. Thus, the benefit of linking the decision of the optimal choice of the current joint acceleration to the resulting effect and similar decision at the preview instant would be completely lost.

As a result, for a more practical formulation that would lead to an effective solution of a joint *LQ* problem, we shall:

1) keep the same constraints at the current instant, to guarantee that $\ddot{\boldsymbol{q}}_k$ realizes the task at $t = t_k$;
2) preserve a forward coupling between the current acceleration and the data/command at the preview instant, allowing a linear dependence of the constraints and a quadratic dependence of the objective function on $\ddot{\boldsymbol{q}}_k$.

Consider again the task constraint (6). We approximate it at time $t = t_{k+1}$ as follows:

$$\begin{aligned} \ddot{\boldsymbol{x}}_{k+1} &= \boldsymbol{J}(\boldsymbol{q}_{k+1})\ddot{\boldsymbol{q}}_{k+1} + \dot{\boldsymbol{J}}(\boldsymbol{q}_{k+1})\dot{\boldsymbol{q}}_{k+1} \\ &\approx \boldsymbol{J}(\boldsymbol{q}_k + \dot{\boldsymbol{q}}_k T)\ddot{\boldsymbol{q}}_{k+1} \\ &\quad + \frac{\boldsymbol{J}(\boldsymbol{q}_k + \dot{\boldsymbol{q}}_k T) - \boldsymbol{J}(\boldsymbol{q}_k)}{T}\,(\dot{\boldsymbol{q}}_k + T\ddot{\boldsymbol{q}}_k) \\ &= \boldsymbol{J}_{k^+}\ddot{\boldsymbol{q}}_{k+1} + (\boldsymbol{J}_{k^+} - \boldsymbol{J}_k)\,\ddot{\boldsymbol{q}}_k + \boldsymbol{h}_{k^+}, \end{aligned} \tag{15}$$

with the notation

$$\boldsymbol{J}_{k^+} = \boldsymbol{J}(\boldsymbol{q}_k + \dot{\boldsymbol{q}}_k T), \quad \boldsymbol{h}_{k^+} = \frac{\boldsymbol{J}_{k^+} - \boldsymbol{J}_k}{T}\,\dot{\boldsymbol{q}}_k. \tag{16}$$

Next, consider the squared norm of the torque at $t = t_{k+1}$. Taking into account the factorization (3) of the quadratic velocity terms, we proceed with the following approximation:

$$\begin{aligned} \|\boldsymbol{\tau}_{k+1}\|^2 &= \|\boldsymbol{M}(\boldsymbol{q}_{k+1})\ddot{\boldsymbol{q}}_{k+1} + \boldsymbol{c}(\boldsymbol{q}_{k+1}, \dot{\boldsymbol{q}}_{k+1}) + \boldsymbol{g}(\boldsymbol{q}_{k+1})\|^2 \\ &= \ddot{\boldsymbol{q}}_{k+1}^T\boldsymbol{M}_{k+1}^2\ddot{\boldsymbol{q}}_{k+1} + 2\left(\boldsymbol{c}_{k+1} + \boldsymbol{g}_{k+1}\right)^T\boldsymbol{M}_{k+1}\ddot{\boldsymbol{q}}_{k+1} \\ &\quad + \left(\boldsymbol{c}_{k+1} + \boldsymbol{g}_{k+1}\right)^T\left(\boldsymbol{c}_{k+1} + \boldsymbol{g}_{k+1}\right) \\ &\approx \ddot{\boldsymbol{q}}_{k+1}^T\boldsymbol{M}_{k^+}^2\,\ddot{\boldsymbol{q}}_{k+1} + 2\left(\boldsymbol{S}_{k^+}\dot{\boldsymbol{q}}_{k+1} + \boldsymbol{g}_{k^+}\right)^T\boldsymbol{M}_{k^+}\,\ddot{\boldsymbol{q}}_{k+1} \\ &\quad + \dot{\boldsymbol{q}}_{k+1}^T\boldsymbol{S}_{k^+}^T\boldsymbol{S}_{k^+}\dot{\boldsymbol{q}}_{k+1} + 2\,\boldsymbol{g}_{k^+}^T\boldsymbol{S}_{k^+}\dot{\boldsymbol{q}}_{k+1} + \boldsymbol{g}_{k^+}^T\boldsymbol{g}_{k^+} \end{aligned}$$

$$
\begin{aligned}
&= \ddot{\boldsymbol{q}}_{k+1}^T \boldsymbol{M}_{k^+}^2 \ddot{\boldsymbol{q}}_{k+1} \\
&+ 2\left(\boldsymbol{S}_{k^+}\left(\dot{\boldsymbol{q}}_k + T\ddot{\boldsymbol{q}}_k\right) + \boldsymbol{g}_{k^+}\right)^T \boldsymbol{M}_{k^+} \ddot{\boldsymbol{q}}_{k+1} \\
&+ \left(\dot{\boldsymbol{q}}_k + T\ddot{\boldsymbol{q}}_k\right)^T \boldsymbol{S}_{k^+}^T \boldsymbol{S}_{k^+}\left(\dot{\boldsymbol{q}}_k + T\ddot{\boldsymbol{q}}_k\right) \\
&+ 2\,\boldsymbol{g}_{k^+}^T \boldsymbol{S}_{k^+}\left(\dot{\boldsymbol{q}}_k + T\ddot{\boldsymbol{q}}_k\right) + \boldsymbol{g}_{k^+}^T \boldsymbol{g}_{k^+},
\end{aligned}
\tag{17}
$$

with the notation

$$
\begin{aligned}
&\boldsymbol{M}_{k^+} = \boldsymbol{M}(\boldsymbol{q}_k + \dot{\boldsymbol{q}}_k T), \quad \boldsymbol{g}_{k^+} = \boldsymbol{g}(\boldsymbol{q}_k + \dot{\boldsymbol{q}}_k T), \\
&\boldsymbol{S}_{k^+} = \boldsymbol{S}(\boldsymbol{q}_k + \dot{\boldsymbol{q}}_k T, \dot{\boldsymbol{q}}_k) = \mathrm{col}\{\dot{\boldsymbol{q}}_k^T \boldsymbol{C}_i(\boldsymbol{q}_k + \dot{\boldsymbol{q}}_k T)\}.
\end{aligned}
$$

With these expressions at hand, the nonlinear optimization problem (14) is replaced by the following *LQ* approximation:

$$
\begin{aligned}
\min_{\ddot{\boldsymbol{q}}_k, \ddot{\boldsymbol{q}}_{k+1}} \quad & H_2 = \tfrac{1}{2}\left(\omega_k \|\boldsymbol{\tau}_k\|^2 + \omega_{k+1}\|\boldsymbol{\tau}_{k+1}\|^2\right) \\
\text{s.t.} \quad & \boldsymbol{\tau}_k = \boldsymbol{M}_k \ddot{\boldsymbol{q}}_k + \boldsymbol{n}_k \\
& \ddot{\boldsymbol{x}}_k = \boldsymbol{J}_k \ddot{\boldsymbol{q}}_k + \boldsymbol{h}_k \\
& \boldsymbol{\tau}_{k+1} = \boldsymbol{M}_{k^+} \ddot{\boldsymbol{q}}_{k+1} + \boldsymbol{S}_{k^+}\left(\dot{\boldsymbol{q}}_k + T\ddot{\boldsymbol{q}}_k\right) + \boldsymbol{g}_{k^+} \\
& \ddot{\boldsymbol{x}}_{k+1} = \boldsymbol{J}_{k^+} \ddot{\boldsymbol{q}}_{k+1} + \left(\boldsymbol{J}_{k^+} - \boldsymbol{J}_k\right)\ddot{\boldsymbol{q}}_k + \boldsymbol{h}_{k^+}.
\end{aligned}
\tag{18}
$$

In particular, replacing the expressions of the two torques $\boldsymbol{\tau}_k$ and $\boldsymbol{\tau}_{k+1}$ in the objective function, the latter takes the form

$$
H_2 = \tfrac{1}{2}\begin{pmatrix} \ddot{\boldsymbol{q}}_k & \ddot{\boldsymbol{q}}_{k+1} \end{pmatrix}^T \boldsymbol{Q} \begin{pmatrix} \ddot{\boldsymbol{q}}_k \\ \ddot{\boldsymbol{q}}_{k+1} \end{pmatrix} + \boldsymbol{r}^T \begin{pmatrix} \ddot{\boldsymbol{q}}_k \\ \ddot{\boldsymbol{q}}_{k+1} \end{pmatrix} + s, \tag{19}
$$

with

$$
\boldsymbol{Q} = \begin{pmatrix} \omega_k \boldsymbol{M}_k^2 + \omega_{k+1} T^2 \boldsymbol{S}_{k^+}^T \boldsymbol{S}_{k^+} & \omega_{k+1} T \boldsymbol{S}_{k^+}^T \boldsymbol{M}_{k^+} \\ \textit{symm} & \omega_{k+1} \boldsymbol{M}_{k^+}^2 \end{pmatrix}, \tag{20}
$$

$$
\boldsymbol{r} = \begin{pmatrix} \omega_k \boldsymbol{M}_k\left(\boldsymbol{S}_k \dot{\boldsymbol{q}}_k + \boldsymbol{g}_k\right) + \omega_{k+1} T \boldsymbol{S}_{k^+}^T\left(\boldsymbol{S}_{k^+}\dot{\boldsymbol{q}}_k + \boldsymbol{g}_{k^+}\right) \\ \omega_{k+1} \boldsymbol{M}_{k^+}\left(\boldsymbol{S}_{k^+}\dot{\boldsymbol{q}}_k + \boldsymbol{g}_{k^+}\right) \end{pmatrix}, \tag{21}
$$

and a scalar s that is irrelevant for the optimization. Similarly, the task-based constraints in (18) can be written in matrix form as

$$
\boldsymbol{A} \begin{pmatrix} \ddot{\boldsymbol{q}}_k \\ \ddot{\boldsymbol{q}}_{k+1} \end{pmatrix} = \boldsymbol{b},
$$

where

$$
\boldsymbol{A} = \begin{pmatrix} \boldsymbol{J}_k & \boldsymbol{0} \\ \boldsymbol{J}_{k^+} - \boldsymbol{J}_k & \boldsymbol{J}_{k^+} \end{pmatrix}, \quad \boldsymbol{b} = \begin{pmatrix} \ddot{\boldsymbol{x}}_k - \boldsymbol{h}_k \\ \ddot{\boldsymbol{x}}_{k+1} - \boldsymbol{h}_{k^+} \end{pmatrix}. \tag{22}
$$

Being $\boldsymbol{Q}$ in (20) positive definite (see Appendix) and provided that matrix $\boldsymbol{A}$ has full (row) rank equal to $2m$, problem (18) is in the form (7) with the positions (20), (21) and (22). Thus, its solution has the closed-form expression (8) with

$$
\begin{aligned}
&\ddot{\boldsymbol{q}}^* = \begin{pmatrix} \ddot{\boldsymbol{q}}_k^T & \ddot{\boldsymbol{q}}_{k+1}^T \end{pmatrix}^T, \\
&\|\boldsymbol{\tau}^*\|^2 = \omega_k \|\boldsymbol{\tau}_k\|^2 + \omega_{k+1}\|\boldsymbol{\tau}_{k+1}\|^2.
\end{aligned}
\tag{23}
$$

This model-based preview solution will be denoted as *MBP*. In the implementation, $\ddot{\boldsymbol{q}}_{k+1}$ is discarded and only $\ddot{\boldsymbol{q}}_k$ from (23) will be used at the instant $t = t_k$. In this case, a feedback control action in the form (12) will be added only inside the term $\ddot{\boldsymbol{x}}_k$ in (22), whereas we shall keep $\ddot{\boldsymbol{x}}_{k+1} = \ddot{\boldsymbol{x}}_{d,k+1}$ for the preview instant.

A. Inclusion of dynamic damping in the null space

In problems (10) and (18), the joint torque realizing the Cartesian task is optimized so that its norm is the closest possible to zero. On the other hand, one could minimize the difference in norm with respect to a suitable desired target torque. Define

$$
\boldsymbol{\tau}_{D_k} = -\boldsymbol{D}_k \boldsymbol{M}_k \dot{\boldsymbol{q}}_k, \tag{24}
$$

where $\boldsymbol{D}_k$ is a non-negative diagonal (damping) gain matrix and $\boldsymbol{M}_k\dot{\boldsymbol{q}}_k$ is the generalized momentum of the robot at $t = t_k$. When $\boldsymbol{\tau}_k = \boldsymbol{\tau}_{D_k}$ and the damping matrix is in the form $\boldsymbol{D}_k = d\boldsymbol{I} > 0$, from (1) and (3), the joint acceleration $\ddot{\boldsymbol{q}}_k$ becomes

$$
\ddot{\boldsymbol{q}}_k = -\boldsymbol{M}_k^{-1}(\boldsymbol{S}_k \dot{\boldsymbol{q}}_k + \boldsymbol{g}_k) - d\dot{\boldsymbol{q}}_k. \tag{25}
$$

Observing eq. (25), the effect of the desired torque (24) is always to work against the current joint velocity, acting thus as a damper on the joint motion of the robot. This will eliminate, or at least reduce, whipping effects and oscillations that happen when the joint velocity becomes too large.

Therefore, the objective function in (10) is modified as

$$
\begin{aligned}
H_3 &= \tfrac{1}{2}\left\|\boldsymbol{\tau}_k - \boldsymbol{\tau}_{D_k}\right\|^2 \\
&= \tfrac{1}{2}\left\|\boldsymbol{M}_k \ddot{\boldsymbol{q}}_k + (\boldsymbol{S}_k + \boldsymbol{D}_k \boldsymbol{M}_k)\dot{\boldsymbol{q}}_k + \boldsymbol{g}_k\right\|^2,
\end{aligned}
\tag{26}
$$

and the joint acceleration solution $\ddot{\boldsymbol{q}}^* = \ddot{\boldsymbol{q}}_k$ will be obtained from (8) by substituting

$$
\begin{aligned}
\boldsymbol{Q} = \boldsymbol{M}_k^2, \qquad & \boldsymbol{r} = \boldsymbol{M}_k((\boldsymbol{S}_k + \boldsymbol{D}_k \boldsymbol{M}_k)\dot{\boldsymbol{q}}_k + \boldsymbol{g}_k), \\
\boldsymbol{A} = \boldsymbol{J}_k, \qquad & \boldsymbol{b} = \ddot{\boldsymbol{x}}_k - \boldsymbol{h}_k.
\end{aligned}
\tag{27}
$$

This solution will be denoted as *MTND* (i.e., *MTN* with damping). The only difference between the expressions (11) and (27), respectively in the *MTN* and *MTND* solutions, is in the $\boldsymbol{r}$ term, namely in an additional damping effect appearing in the null space of the task Jacobian.

In case of torque optimization using model-based preview (18), in place of (26) we consider the objective function

$$
H_4 = \tfrac{1}{2}\left(\omega_k \|\boldsymbol{\tau}_k - \boldsymbol{\tau}_{D_k}\|^2 + \omega_{k+1}\|\boldsymbol{\tau}_{k+1} - \boldsymbol{\tau}_{D_{k^+}}\|^2\right), \tag{28}
$$

in which $\boldsymbol{\tau}_{D_{k^+}} = -\boldsymbol{D}_{k+1}\boldsymbol{M}_{k^+}\left(\dot{\boldsymbol{q}}_k + T\ddot{\boldsymbol{q}}_k\right)$. The optimal solution $\ddot{\boldsymbol{q}}^* = \left(\ddot{\boldsymbol{q}}_k^T \;\; \ddot{\boldsymbol{q}}_{k+1}^T\right)^T$ is given again by (8), using for $\boldsymbol{A}$ and $\boldsymbol{b}$ the same substitutions (22) while replacing in the expressions (20–21) the $\boldsymbol{S}_k$ term in $\boldsymbol{r}$ with $(\boldsymbol{S}_k + \boldsymbol{D}_k\boldsymbol{M}_k)$ and each $\boldsymbol{S}_{k^+}$ in $\boldsymbol{r}$ and $\boldsymbol{Q}$ with $(\boldsymbol{S}_{k^+} + \boldsymbol{D}_{k+1}\boldsymbol{M}_{k^+})$. The resulting solution is denoted as *MBPD*. A feedback control action on the task error is included as in the undamped cases.

IV. Simulation results

A. 3R planar arm

To illustrate the comparison between the different solutions for local torque optimization, we have considered a 3R planar arm ($n = 3$) moving on a horizontal plane and the same Cartesian trajectories $\boldsymbol{x}_d(t)$ presented in [1]. The robot has links of equal length $l = 1$ [m], uniformly distributed mass $m_l = 10$ [kg] and moment of inertia $I_l = m_l l^2/12$. The end-effector position ($m = 2$) should follow a linear path of short length ($L_1 = 0.2828$ [m]), or one four times longer

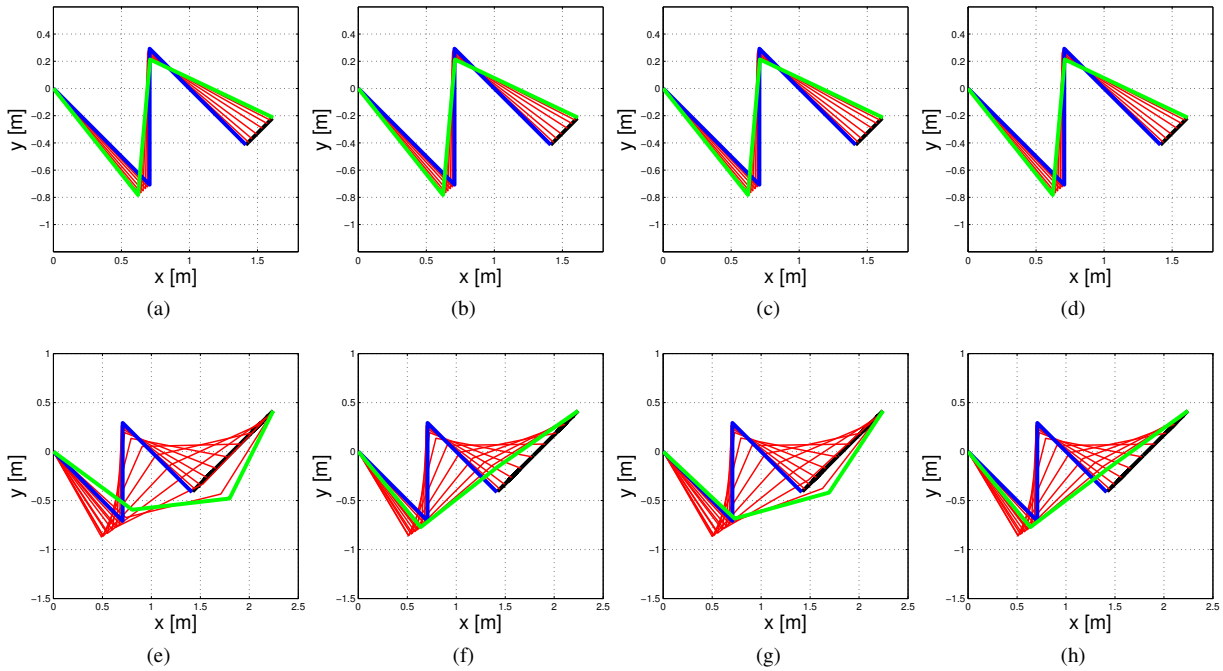

Fig. 1: Stroboscopic motion of the 3R planar arm on a short [top] and a long [bottom] Cartesian path using different torque optimization methods: (a-e) *MTN*; (b-f) *MTND*; (c-g) *MBP*; (d-h) *MBPD*. The desired linear paths are in black. Initial and final arm configurations are shown in blue and green, respectively.

($L_2 = 1.1738$ [m]). The path is traced with a rest-to-rest timing law having bang-bang acceleration of magnitude $A = \sqrt{2} = 1.4142$ [m/s^2]. The robot starts with $\dot{\boldsymbol{q}}(0) = \boldsymbol{0}$. The degree of robot redundancy is $n - m = 1$ in this case.

The methods have been implemented in MATLAB with a fixed integration step $T_s = 0.001$ [s]. The *MBP* and *MBPD* methods use equal weights $\omega_k = \omega_{k+1} = 1$ in the objective function. The joint damping matrix is chosen as $\boldsymbol{D}_k = 10\,\boldsymbol{I}$, constant at all discrete times for both *MTND* and *MBPD* methods. In all methods, the feedback gains on the Cartesian task error were $\boldsymbol{K}_P = 10\,\boldsymbol{I}$ and $\boldsymbol{K}_D = \boldsymbol{I}$. The short preview window can be chosen directly as the next sampling instant, i.e., $T = T_s$. More in general, the optimization process can be used to minimize instead the norm of the torques at the current instant and at any other instant in the short future, i.e., $T > T_s$. For longer T, it is expected that the accuracy of the robot state estimation decreases. When $T = 0$, the *MBP* method collapses into the original *MTN* one.

Figure 1 shows stroboscopic views for the four addressed methods along the short and the long path, respectively. In this case, the preview window was $T = 100T_s = 0.1$ [s] for the *MBP* and *MBPD* methods. All optimization methods performed in the same way on the short path, with a rather symmetric behavior with respect to the half-motion time[1]. All methods completed the task with zero final joint velocities in practice, although this was not an explicit constraint in the optimization problem.

Figure 2 shows the norms of the joint velocity and torque on the longer path for the different methods. While a common behavior is found in the first half of the motion, i.e., until $T_m = \sqrt{L_2/A} \approx 0.91$ [s], each method completes the task in a different way during the deceleration phase. With the *MTN* method, it is clear how the contentious increase of the joint velocities leads to the sudden and high peak in the joint torques near the end of the task. Using instead our proposed methods this undesired behavior is eliminated, with the *MBP* method having the minimum values for the joint torques norm. The *MTND* and *MBPD* methods produced almost the same behavior, with the joint velocities vanishing at the end of the task. The Cartesian task error was in all cases negligible, in the order of 10^{-4} [m].

Figure 3 shows the joint velocity and torque norms obtained in multiple simulations on the long linear task, when using the *MBP* solution with different preview values T. Too short values, e.g., $T = T_s = 1$ [ms], had no practical effect and returned the same bad behavior of the *MTN* method. For an intermediate set of preview values, i.e., $T \in [0.1, 0.9]$ s, the robot loses the undesired torque peak and the behaviors will be similarly good. For the *MBP* method, a preview of $T = 100\,T_s = 0.1$ [s] gives the best result in terms of torque norm, while $T = 900\,T_s = 0.9$ s gives the best result for the joint velocity norm. In general, joint velocities become lower and torques higher when increasing T, until a limit is reached where the peaks appear back (here, for $T = 1000\,T_s = 1$ [s]) and robot motion is unacceptable again. Indeed, the best choice for T will depend on the desired trajectory and on the dynamics of the specific robot. Nonetheless, for a given robot/trajectory pair, the existence of an interval of good performance for the preview method appears to be robust.

[1]The plots are not reported due to lack of space, but joint motions and torques in the first half of the trajectory looks very similar to those of Fig. 2.

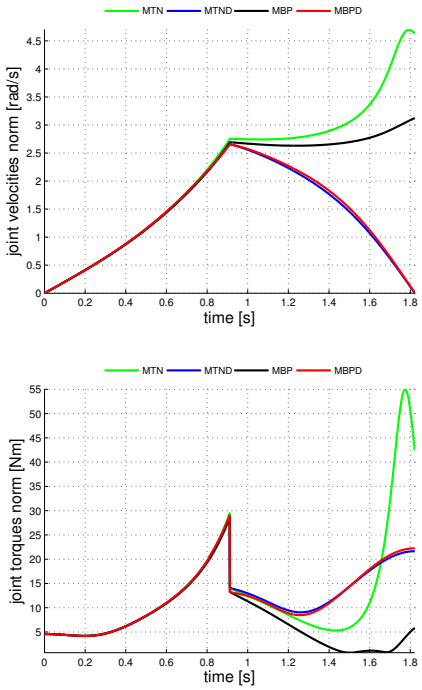

Fig. 2: Joint velocity norms [top] and torque norms [bottom] for the 3R planar arm using different solutions on the long trajectory.

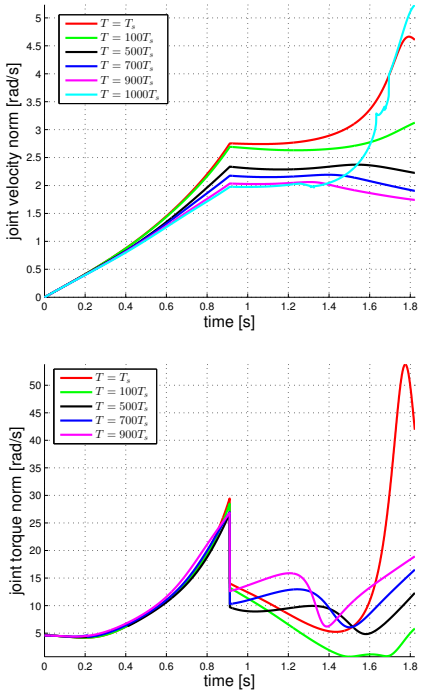

Fig. 3: Joint velocity norms [top] and torque norms [bottom] for the 3R planar arm on the long trajectory with the *MBP* method using different preview windows T. Simulations run at $T_s = 1$ [ms]. The torque norm for $T = 1000T_s = 1$ [s] is not shown being too large.

B. KUKA LWR robot

As a second case study, we considered a 7R KUKA LWR robot and performed simulations using V-REP and our C++ code. In the integration routine, we used $T_s = 10$ [ms] as the fixed step. This value is also the lowest refresh rate for V-REP. The obtained results are not reported here because they were almost identical to those obtained in the experiments of Sec. V. In any event, plots of the relevant variables for the different methods can be found in the accompanying video.

The simulation environment allowed us to test again the performance of the *MBP* method for different preview windows T. Figure. 4 shows in particular the behavior of the joint velocity norm along a long linear path. As expected, large oscillations are found for $T = 0$ (the *MTN* method). Between $T = T_s = 10$ [ms] and $T = 10\,T_s = 100$ [ms], the *MBP* method works fine. For $T = 15\,T_s = 150$ [ms] or larger, the joint velocity norm becomes unacceptable again, i.e., much larger and oscillatory in nature.

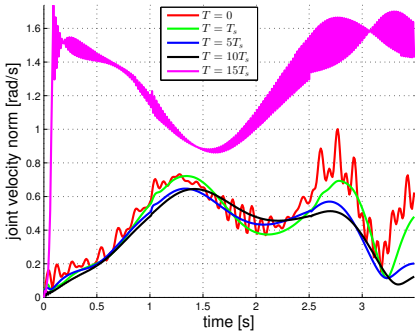

Fig. 4: Joint velocity norm for the KUKA LWR robot with the *MBP* method using different preview windows T (simulations in V-REP with $T_s = 10$ [ms]).

V. Experiments with a KUKA LWR robot

For the 7R KUKA LWR 4 robot in our lab, we considered as task the execution of two Cartesian trajectories defined for the position of its end-effector flange center ($m = 3$). Since the rotation of joint 7 has no effect on this task, the last joint was frozen resulting in only $n = 6$ active dofs for the robot, and a degree of redundancy $n - m = 3$. All model-based computations were done using the accurate dynamic model identified in [19]. The timing law prescribes a rest-to-rest motion on the path with a trapezoidal speed profile. The robot starts with $\dot{\boldsymbol{q}}(0) = \mathbf{0}$.

The different torque optimization methods were implemented using C++. Experiments were performed using the position control mode of the LWR through the KUKA FRI library, feeding as reference the instantaneous motion obtained from the torque optimization schemes, with a sampling time $T_s = 5$ [ms]. The torques measured by the joint torque sensors during task execution are used to assess the obtained performance. Torque data have been processed through a low-pass filter to eliminate measurement noise. The complete results are shown in the experimental part of the video accompanying the paper.

The first motion task is on a relatively short linear path of length $L = 0.5$ [m] to be traced in $T_f = 3.5$ [s]. The second task is longer and consists in following a circular path of radius $R = 0.2$ [m] for an arc length $L_c = 1.256$ [m] and a motion time $T_f = 7.28$ [s]. For both tasks, the desired maximum Cartesian velocity and acceleration were chosen as $V = 0.2$ [m/s] and $A = 0.2$ [m/s^2], respectively.

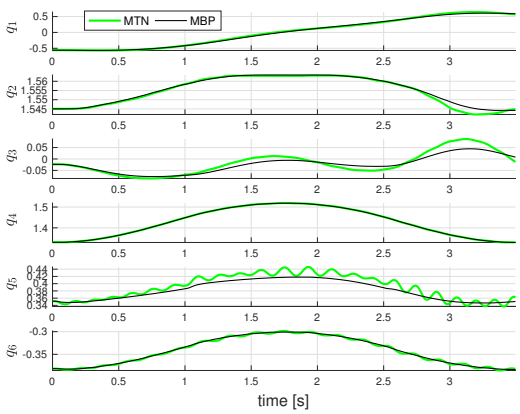

Fig. 5: Experiment along a linear path. Joint positions for the KUKA LWR using the *MTN* and *MBP* methods.

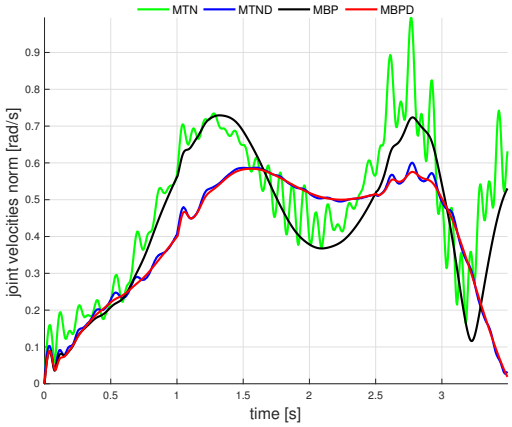

Fig. 6: Experiment along a linear path. Joint velocity norms for the KUKA LWR using different optimization methods.

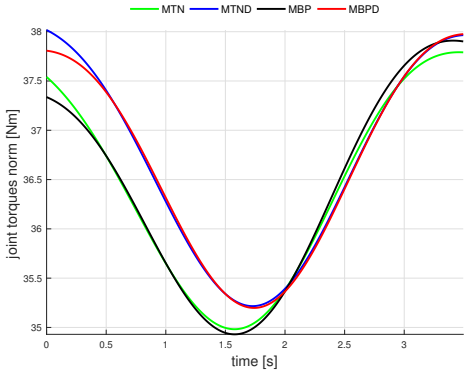

Fig. 7: Experiment along a linear path. Joint torque norms for the KUKA LWR using different optimization methods.

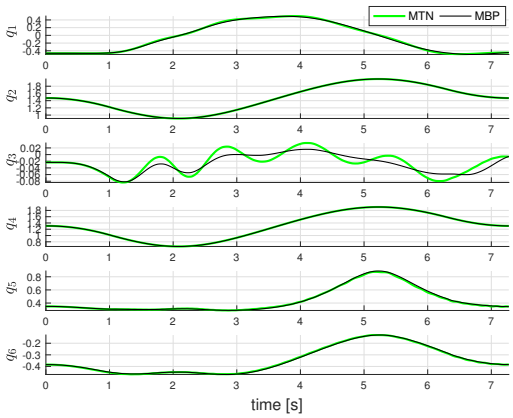

Fig. 8: Experiment along a circular path. Joint positions for the KUKA LWR using the *MTN* and *MBP* methods.

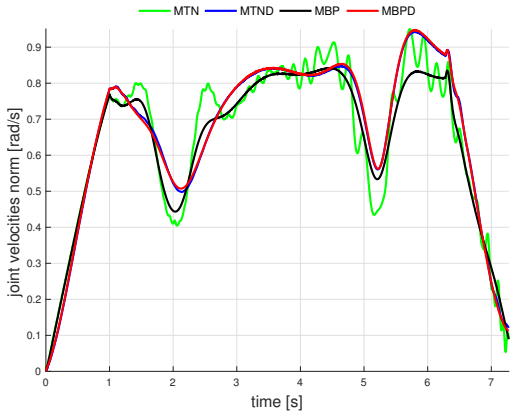

Fig. 9: Experiment along a circular path. Joint velocity norms for the KUKA LWR using different optimization methods.

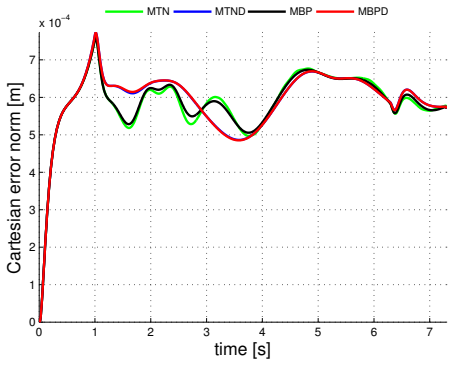

Fig. 10: Cartesian error norm along the circular path in the KUKA LWR experiments using different torque optimization methods.

The weights in the objective function of both *MBP* and *MBPD* methods were $\omega_k = \omega_{k+1} = 1$. The damping matrix was constantly $\boldsymbol{D}_k = 5\,\boldsymbol{I}$ for the *MTND* and *MBPD* methods, while $\boldsymbol{D}_{k+1} = \boldsymbol{I}$ was used in the *MBPD* method. In all methods, the feedback gains on the Cartesian task error were $\boldsymbol{K}_P = 300\,\boldsymbol{I}$ and $\boldsymbol{K}_D = 50\,\boldsymbol{I}$. The preview window in the experiments was set to $T = T_s = 5$ [ms].

Figures 5–7 show the results obtained in the experiments along the linear path. The fast oscillatory behavior of the joint velocity norm generated by the *MTN* method during most part of the motion is quite evident in Fig. 6. More specifically, this instability appears mostly in the fifth joint position, but also in the third and sixth joint profiles (see Fig. 5). It should be noted that the evolutions of the torque norms in Fig. 7 differ only slightly between the four methods, and the mean torque norms over the entire motion, i.e., $\frac{1}{T_f}\int_0^{T_f} \|\boldsymbol{\tau}\|_2\, dt$, are all very similar (their numerical values remain all around 36.6 [Nm]). In contrast, the proposed *MBP* and *MBPD* methods eliminate any undesired behavior, both in the joint velocities and in the torques. Both damped methods achieve better results in forcing

the joint velocities toward zero at the end. On the other hand, the *MTND* method has still some residual oscillations in the joint velocity.

Figures 8–10 reports the results for the experiments along the circular path. The third joint position in Fig. 8 experiences the main oscillations in this case, while Fig. 9 shows a clear increasing trend of these in the joint velocity norm when using the *MTN* method. The *MBP*, *MBPD* and *MTND* methods eliminate any undesired behavior. Similarly to Fig. 7, the evolutions of the torque norms (not reported) differ again very slightly between the four methods. In both tasks, joint oscillations do not affect the execution of the desired Cartesian trajectory, also thanks to the presence of the feedback control action on the task error. The maximum of the Cartesian error norm in Fig. 10 is about 7×10^{-4} [m] for the circular path (and was even less for the linear path).

VI. Conclusions

Different torque optimization methods have been proposed to address the instability issue in redundant robots when the norm of the joint torque is instantaneously minimized. The model-based short preview scheme *MBP* optimizes the norm of the joint torque at the current and at a single future but close instant, anticipating the possible dramatic growth of joint torques. Alternatively, the introduction of a desired momentum-damping joint torque in the null space of the task can reduce the associated drift in joint velocities. These two local control schemes can be used separately (*MTND*) or in combination (*MBPD*), leading to robot behaviors that are consistently stable in performing short and long task trajectories, without peaks or oscillations in torques or velocities. In general, *MBPD* seems the best optimization method since it generates smooth motion with lower residual joint velocity at the end and with a similar torque demand of the other methods. Despite the selection of a best preview instant is likely to depend on the desired task and on the robot own dynamics, we found in all cases a favourable insensitivity of the achieved robot performance when the preview instant was chosen in a non-vanishing intermediate range.

Future work will address an adaptive choice of the preview window, as well as the use of different weights for the torques at the two instants considered in the optimization scheme with short preview. The preview concept and the momentum-damping technique could be used also for other types of dynamic optimization problems (e.g., minimum energy).

Appendix

We provide here a simple proof that the matrix $\boldsymbol{Q}$ in (20) for the *MBP* method is always symmetric and positive definite. The symmetry of $\boldsymbol{Q}$ follows immediately from its construction and being the squared inertia matrix $\boldsymbol{M}^2(\boldsymbol{q})$ itself symmetric. A symmetric matrix $\boldsymbol{Q}$ is positive definite iff $\boldsymbol{v}^T\boldsymbol{Q}\,\boldsymbol{v} > 0$, $\forall \boldsymbol{v} \neq \boldsymbol{0}$. Splitting $\boldsymbol{v} \neq \boldsymbol{0}$ in two parts $\boldsymbol{v}_1$ and $\boldsymbol{v}_2$ according to the block matrix structure in (20), we have

$$\begin{aligned}\boldsymbol{v}^T\boldsymbol{Q}\,\boldsymbol{v} &= \begin{pmatrix}\boldsymbol{v}_1^T & \boldsymbol{v}_2^T\end{pmatrix} \boldsymbol{Q} \begin{pmatrix}\boldsymbol{v}_1 \\ \boldsymbol{v}_2\end{pmatrix} \\ &= \omega_k ||\boldsymbol{M}_k \boldsymbol{v}_1||^2 + \omega_{k+1} ||\boldsymbol{M}_{k^+}\boldsymbol{v}_2 + T\boldsymbol{S}_{k^+}\boldsymbol{v}_1||^2.\end{aligned}$$

Since $\boldsymbol{M}(\cdot)$ is positive definite for all its possible arguments, then both $\boldsymbol{M}_k$ and $\boldsymbol{M}_{k^+}$ will be positive definite. Thus, $\boldsymbol{M}_k\boldsymbol{v}_1 \neq \boldsymbol{0}$ and $\boldsymbol{M}_{k^+}\boldsymbol{v}_2 \neq \boldsymbol{0}$ for all $\boldsymbol{v}_i \neq \boldsymbol{0}$, $i = 1, 2$. Now consider the two cases:

- $\boldsymbol{v}_1 = \boldsymbol{0}, \boldsymbol{v}_2 \neq \boldsymbol{0}$; then $\boldsymbol{v}^T\boldsymbol{Q}\,\boldsymbol{v} = \omega_{k+1}||\boldsymbol{M}_{k^+}\boldsymbol{v}_2||^2 > 0$, since also $\omega_{k+1} > 0$;
- $\boldsymbol{v}_1 \neq \boldsymbol{0}$; being $\omega_k > 0$, the term $\omega_k||\boldsymbol{M}_k\boldsymbol{v}_1||^2 > 0$; since $\omega_{k+1} > 0$ and the norm of a vector is always non-negative, then $\boldsymbol{v}^T\boldsymbol{Q}\,\boldsymbol{v} > 0$ also in this case.

The same procedure can be used to prove the symmetry and positive definiteness of the $\boldsymbol{Q}$ matrix in the *MBPD* method.

References

[1] J. Hollerbach and K. Suh, "Redundancy resolution of manipulators through torque optimization," *IEEE J. of Robotics and Automation*, vol. 3, no. 4, pp. 308–316, 1987.

[2] ——, "Local versus global torque optimization of redundant manipulators," in *Proc. IEEE Int. Conf. on Robotics and Automation*, 1987, pp. 619–624.

[3] K. Kazerounian and Z. Wang, "Global versus local optimization in redundancy resolution of robotic manipulators," *Int. J. of Robotics Research*, vol. 7, no. 5, pp. 3–12, 1988.

[4] I.-C. Shim and Y.-S. Yoon, "Stabilization constraint method for torque optimization of a redundant manipulator," in *Proc. IEEE Int. Conf. on Robotics and Automation*, 1997, pp. 2403–2408.

[5] S. Ma, "Local torque minimization of redundant manipulators with considering end-motion joint velocities," in *Proc. IEEE Int. Conf. on Systems, Man, and Cybernetics*, 1996, pp. 1477–1482.

[6] S. Ma and D. N. Nenchev, "Local torque minimization for redundant manipulators: A correct formulation," *Robotica*, vol. 14, no. 2, pp. 235–239, 1996.

[7] T.-H. Chen, F.-T. Cheng, Y.-Y. Sun, and M.-H. Hung, "Torque optimization schemes for kinematically redundant manipulators," *J. of Robotic Systems*, vol. 11, no. 4, pp. 257–269, 1994.

[8] S. Ma, "A balancing technique to stabilize local torque optimization solution of redundant manipulators," *J. of Robotic Systems*, vol. 13, no. 3, pp. 177–185, 1996.

[9] K. O'Neil and Y.-C. Chen, "Instability of pseudoinverse acceleration control of redundant mechanisms," in *Proc. IEEE Int. Conf. on Robotics and Automation*, 2000, pp. 2575–2582.

[10] K. O'Neil, "Divergence of linear acceleration-based redundancy resolution schemes," *IEEE Trans. on Robotics and Automation*, vol. 18, no. 4, pp. 625–631, 2002.

[11] J. Peters, M. Mistry, F. Udwadia, J. Nakanishi, and S. Schaal, "A unifying framework for robot control with redundant DOFs," *Autonomous Robots*, vol. 24, no. 1, pp. 1–12, 2008.

[12] A. Herzog, N. Rotella, S. Mason, F. Grimminger, S. Schaal, and L. Righetti, "Momentum control with hierarchical inverse dynamics on a torque-controlled humanoid," *Autonomous Robots*, vol. 40, no. 3, pp. 473–491, 2016.

[13] E. Mingo Hoffman, A. Laurenzi, L. Muratore, N. G. Tsagarakis, and D. G. Caldwell, "Multi-priority Cartesian impedance control based on quadratic programming optimization," in *Proc. IEEE Int. Conf. on Robotics and Automation*, 2018, pp. 309–315.

[14] V. Duchaine, S. Bouchard, and C. M. Gosselin, "Computationally efficient predictive robot control," *IEEE/ASME Trans. on Mechatronics*, vol. 12, no. 5, pp. 570–578, 2007.

[15] P. Poignet and M. Gautier, "Nonlinear model predictive control of a robot manipulator," in *Proc. 6th Int. Work. on Advanced Motion Control*, 2000, pp. 401–406.

[16] Y. Tassa, T. Erez, and E. Todorov, "Synthesis and stabilization of complex behaviors through online trajectory optimization," in *Proc. IEEE/RSJ Int. Conf. on Intelligent Robots and Systems*, 2012, pp. 4906–4913.

[17] N. Scianca, M. Cognetti, D. De Simone, L. Lanari, and G. Oriolo, "Intrinsically stable MPC for humanoid gait generation," in *Proc. 16th IEEE-RAS Int. Conf. on Humanoid Robots*, 2016, pp. 601–606.

[18] D. G. Luenberger and Y. Ye, *Linear and Nonlinear Programming*, 3rd ed. Springer, 2010.

[19] C. Gaz, F. Flacco, and A. De Luca, "Identifying the dynamic model used by the KUKA LWR: A reverse engineering approach," in *Proc. IEEE Int. Conf. on Robotics and Automation*, 2014, pp. 1386–1392.